A utilização da derivada no processamento de imagem

Victor Costa da Silva Campos[2],Gustavo Tupini Silveira[1], Jamisson Jader Moraes Pereira Júnior[1], Matheus de Paula[1], Thiago da Silva Figueiredo, Rangel Henrique Miranda Trindade[1], Vinicius Fernandes Silva[1]

[1]Discentes Instituto de Ciências Exatas e Aplicadas– Universidade Federal de Ouro Preto (UFOP) – 35.931-008 – João Monlevade – MG – Brasil

[2]Docente do Departamente de Engenharia Elétrica – Universidade de Ouro Preto João Monlevade, BR.

Abstract. *The current paper demonstrates a practical application of the derivative calculus in the Computer Engineering area. Based on a segment of great importance in the development of technologies, this project aimed at studying image processing, more precisely the edge detection proposed by John Canny, to demonstrate to students who are in the firsts period the importance of learning differentiation during its formation. It is hoped that students who have already developed a notion of calculus can understand in a simple way the area approached. The process for edge extraction, known as the Canny method, uses the variation of the pixel intensity in an image to generate an image composed only of edges. In addition of exposing the process involved in this procedure, the results of an algorithm developed to apply the image resulted from the edge detection method to perform measurements of an object in real time will be presented. This project gave a hands-on look at the actual application of derivative calculus. Moreover, it made possible knowing an area where there are a many researches and developments to be applied in new projects.*

Resumo. *O presente trabalho demonstra de forma prática a aplicação do cálculo de derivadas na área da Engenharia da Computação. Baseado em um segmento de grande importância no desenvolvimento de tecnologias, este projeto visou estudar o processamento de imagens, mais especificamente a detecção de bordas proposta por John Canny, para demonstrar aos alunos que cursam o primeiro ou segundo período a importância do estudo da diferenciação durante a formação. Espera-se que estudantes que já desenvolveram uma noção do cálculo possam compreender de forma simples a área abordada. O processo para extração de bordas conhecido como método Canny, utiliza a variação da intensidade do pixel de uma imagem para gerar uma outra composta apenas por bordas. Em complemento à exposição dos passos envolvidos nesse procedimento, serão apresentados os resultados de um algoritmo desenvolvido para aplicar a imagem resultante do método de detecção de bordas para realizar medições de um objeto em tempo real. Este projeto proporcionou uma visualização prática da real aplicação do cálculo de derivadas. Ademais, foi possível conhecer uma área onde há muito a se pesquisar e a se desenvolver em novos projetos.*

1. Introdução

A derivada tem como conceito teórico fornecer a variação instantânea de uma função em relação às variáveis em um dado momento. Geometricamente a derivada fornece uma reta tangente, tal que seu ângulo indica o quanto o ponto está variando em relação aos seus vizinhos. O estudo da derivada no primeiro período é necessário para os alunos de qualquer área da engenharia, porém algumas dificuldades podem surgir por conta da falta de demonstração das aplicações do método de diferenciação.

Mesmo com um conceito muito importante, aplicar derivada em situações reais não é algo trivial. Normalmente as situações reais possuem diversas variáveis que impossibilitam chegar a uma fórmula simplificada como é visto em sala de aula nos primeiros períodos da engenharia. Isto significa que existe uma dificuldade em demonstrar a aplicação do cálculo de derivada na área de computação para os alunos do primeiro ano em Engenharia de Computação da UFOP. Entretanto, existem algumas áreas em que é possível ver a aplicação da derivada e ter uma base sobre os mecanismos computacionais que a operam.

A área a ser abordada por esse trabalho deve seguir os seguintes parâmetros: (1) Não exigir conhecimento que ultrapasse a grade curricular do primeiro e segundo período; (2) Ser visível sua aplicação; (3) Não demandar uma quantidade grande de tempo para sua execução. Na engenharia de computação há um método que simplifica a compreensão do aluno diante da derivada, esse denota-se por processamento de imagem, que utiliza a diferenciação em seus cálculos.

O processamento de uma imagem tende a ser um procedimento de entrada e saída de dados que consiste na transformação de uma imagem realçando os fatores de interesse. Por trás desse processo ocorrem vários outros que não podem ser visualizados em um primeiro instante sem que seja feito um estudo sobre o tema, um desses processos é a derivada, onde muitos alunos a utilizam somente para fins matemáticos e não sabem sobre sua importância em diversos seguimentos.

A detecção de borda é uma das vertentes do processamento de imagem, sendo essa a área a ser trabalhada no desenvolvimento desse projeto. Na internet existem diversas bibliotecas que podem ser implementadas em algoritmos sem que o aluno desenvolva técnicas muito avançadas na programação. Ademais, o método Canny é um conceito consolidado, por isso softwares que trabalham com o processamento de imagens já possuem funções próprias para a sua aplicação, o que demanda menor tempo na elaboração do algoritmo. Outra razão da escolha desse processo explica-se pelo fato de as etapas que envolvem a detecção de borda e sua aplicação serem de fácil percepção pelos alunos. Além disso, os conceitos aplicados não exigem conhecimento que ultrapasse o que é demandado no primeiro e segundo período.

Esse trabalho tem como objetivo geral compreender a aplicação de derivada na Engenharia da Computação. Sobretudo, será abordado a área de processamento de imagens e extração de padrões com foco na detecção de borda utilizando o método Canny. Esse objetivo pode ser detalhado nos seguintes objetivos específicos: (1) Compreender a utilização do método Canny para a detecção de borda; (2) Compreender o cálculo da derivada aplicada no método; (3) Desenvolver um algoritmo que detecta bordas em uma imagem; (4) Aprimorar o algoritmo para realizar medições de objetos; Por fim, (5) Im-

plementar o algoritmo para medir objetos em tempo real.

2. Análise crítica

Na execução deste trabalho desde o princípio, houve grande discussão de como seria abordado o contexto de derivada em uma forma prática e aplicável para os alunos, várias vertentes foram apresentadas e discutidas até que foi chegado a um consenso que o tema de pesquisa seria o processamento de imagens, no qual seria demonstrada a detecção de bordas pelo método do Canny. Após a realização da pesquisa para o fundamento do trabalho começaram a surgir empecilhos, visto que seria necessário o entendimento de áreas e plataformas e linguagens de programação que ainda não havia sido abordado para alunos do primeiro período do curso Engenharia da Computação.

Os problemas iniciais surgiram na criação do algoritmo, introduzido a princípio em Scilab e posteriormente alterado para C, pois era a linguagem que estava a ser lecionado no período decorrente, em seguida o problema maior encontrado foi o entendimento de Canny na aplicação da derivada no contexto de detecção de bordas, para este foram recorridos estudos em artigos e buscas aprofundadas sobre o contexto. Com o encerramento destas etapas e o conhecimento empreendido em relação ao tema, chegou-se a conclusão que o projeto poderia ir adiante, utilizado a detecção de bordas para medições de objetos em tempo real, tendo em conhecimento, esse processo já muito utilizado em diversas áreas do nosso cotidiano.

Chegada à finalização do trabalho, foi possível absorver um vasto conhecimento de algumas áreas de atuação e aplicação do curso Engenharia da Computação, o implemento de conhecimentos e dificuldade da apresentação da atuação da derivada para alunos de primeiro e segundo período, sendo esta não só limitada a estes, mas a princípio em maior ênfase.

3. Aplicabilidade do cálculo da engenharia

O estudo do cálculo representa grande avanço na matemática visto que antes do seu aperfeiçoamento estávamos restringidos a aplicações estáticas e menos dinâmicas. O desenvolvimento do cálculo diferencial e integral, possibilitou que diversos problemas de difícil solução se tornassem passíveis de resoluções menos complexas. Com o conceito de modelagem matemática foi possível entender fenômenos por meio da aproximação de um problema para encontrar uma solução exata. Desde então, a aplicação do cálculo tem sido uma técnica de extrema importância.

Com a contribuição de vários estudiosos, é notável que a aplicação do cálculo potencializou descobertas em diversas áreas. Arquimedes, Fermat e Kepler são pensadores da antiguidade que contribuíram para a área, mas somente no século XVII que Isaac Newton e Gottfried Leibniz desenvolveram simultaneamente os conceitos de cálculo diferencial e integral, e acabaram se envolvendo no que ficou conhecida como A Guerra do Cálculo [Dias 2016]. Com as descobertas daquela época foi possível compreender o movimento dos planetas, algumas forças físicas como magnetismo, eletricidade, fluxo de fluídos e o movimento de corpos dentro e fora do globo. Mas não é somente no estudo da física que houve avanço, a compreensão do crescimento de organismos, propagação de doenças e até flutuação da economia foram potencialmente evoluídos com a contribuição do cálculo.

A utilização do cálculo diferencial e integral é, frequentemente, abordado de maneira puramente matemática, sem que haja preocupação de se aprofundar no potencial que isso apresenta para aumentar o interesse dos estudantes de forma a motivar a busca por aplicações práticas. Ao reduzir o ensino a contínuas operações algébricas, é provável que a sua aceitação como disciplina prática de Engenharia seja, aos poucos, prejudicada.

4. Referencial teórico

Para o entendimento desse projeto é necessário revisar alguns conceitos na área da computação, sobretudo o processamento de imagem com foco na detecção de borda.

Dessa forma, pretende-se introduzir os conceitos sobre o processamento de imagem, apresentando como é armazenada uma imagem no computador e o método utilizado pelo grupo na detecção de borda. Por fim, são exibidos os conceitos sobre programação e sobre a biblioteca utilizada para desenvolver o algoritmo do projeto.

4.1. Processamento de imagem

O processamento de imagem atua na influência da iluminação refletida pela superfície de corpos com o objetivo de fazer análises sobre esse. Utilizando imagens é possível extrair informações sobre as características físicas e geométricas básicas dos objetos, tais como a dimensão, a área, o perímetro e a forma. O processamento de imagem é utilizado para extrair essas características e assim encontrar padrões que possibilitem maior conhecimento sobre o corpo.

Segundo [de Queiroz and Gomes 2006] uma imagem monocromática, isto é que possui uma única cor, é uma função contínua bidimensional dada por F (x, y), tal que x e y são as coordenadas espaciais dos pontos da fotografia e F representa a intensidade da luz no local. Para um computador processar imagens é necessário armazenar cada ponto, denominados como *Picture element* (*pixel*), dessas em um *array* bidimensional, ou seja, em uma matriz, por meio de um processo conhecido como amostragem [Silva 2000]. Esse tipo de armazenamento tem como vantagem a possibilidade de aplicar operações matriciais sobre a fotografia, como soma, multiplicação por escalar e etc. Nesse caso, os *pixels* possuem representações no R3: as coordenadas x e y que correspondem sua posição na imagem, e a intensidade de cor no eixo z, veja a imagem 1.

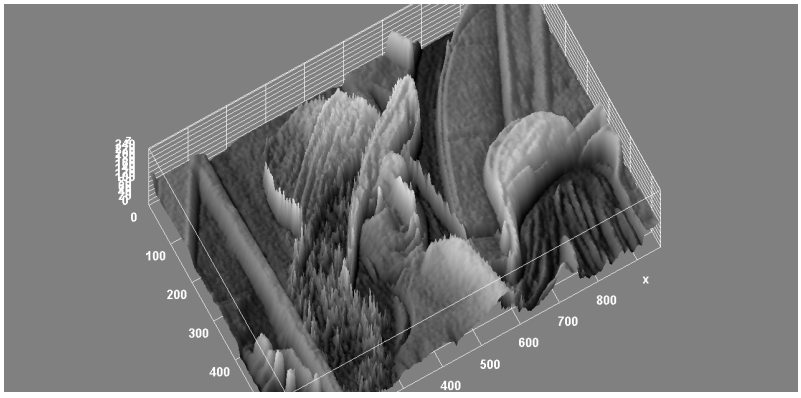

Figura 1. Representação das dimensões da imagem e a intensidade de cada pixel

Normalmente as imagens são utilizadas no sistema *Red, Green and Blue* (RGB), isso significa que cada pixel é uma aproximação resultante das intensidades de vermelho, azul e verde. A união das três cores no pixel forma a imagem de 3 bandas [Biasi et al. 2002]. Segundo [de Queiroz and Gomes 2006] a demonstração da imagem de três bandas pode ser a seguinte equação:

$$F(x, y) = F_r(x, y) + F_g(x, y) + F_b(x, y) \quad (1)$$

A imagem 2 demonstra as bandas de uma imagem colorida.

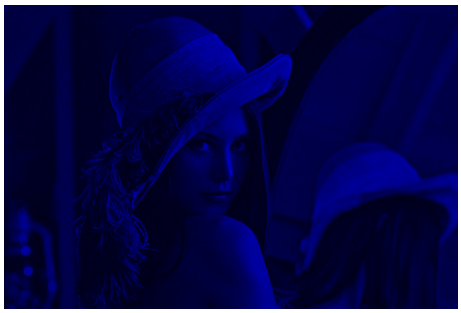

Figura 2. Representação da banda azul

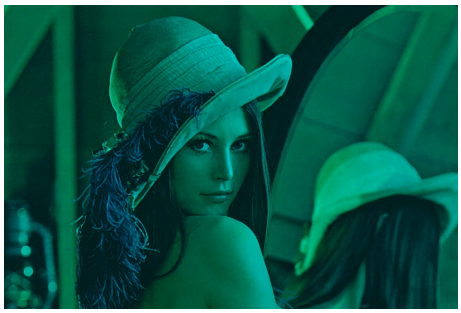

Figura 3. Representação da banda verde

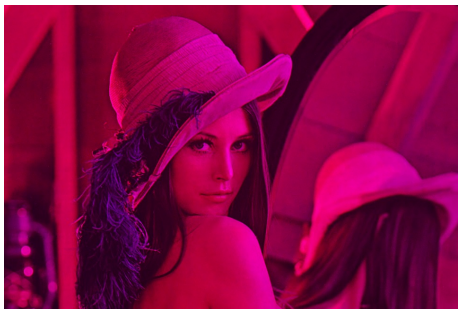

Figura 4. Representação da banda vermelha

Majoritariamente as imagens utilizadas no trabalho foram de 8 bits, ou seja, elas suportam tonalidades de cores nas escalas RGB que variam de 0 a 255, totalizando 2^8 tipos de cores. Segundo [Morgan et al. 2008] O tamanho de um arquivo digital está diretamente relacionado com a qualidade dos detalhes visíveis e o tamanho do intervalo de cores,

entretanto é importante ressaltar que o processamento de arquivos grandes demanda maior tempo, podendo ser inviável sua aplicação. Por convenção na área de computação visual adotou-se que a intensidade do pixel igual a 0 representaria o preto e o 255 representaria o branco, todas as outras cores são variações nesse intervalo.

Segundo [de Queiroz and Gomes 2006] existem duas classificações mais relevantes quando se trata de operações que manipulam de forma direta o pixel: as operações pontuais e as operações locais ou por máscaras. O objetivo desses cálculos é, por meio de uma sequência de funções aplicadas sobre os *pixels* da fotografia original, criar uma imagem de saída que terá alguns parâmetros modificados. Nas operações pontuais cada elemento da imagem da saída estará em função apenas de um único elemento correspondente na imagem de entrada. Entretanto, nas operações locais o valor de saída é dependente não apenas do elemento correspondente na imagem de origem, mas também a vizinhança desse *pixel*, isto é, os elementos que estão dentro da máscara. As máscaras são matrizes com dimensões pequenas que podem ser aplicadas em cada pixel da imagem [Silva 2000]. A imagem a seguir exemplifica essa operação:

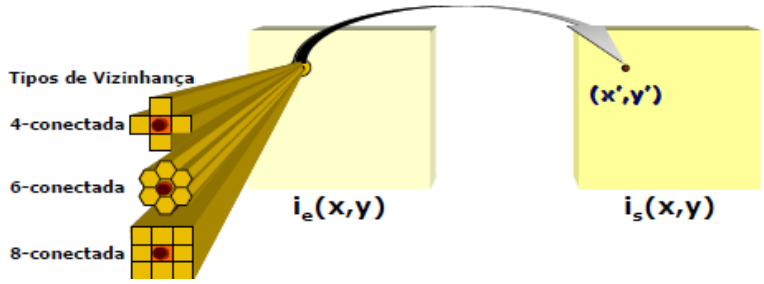

Figura 5. Exemplo dos diagramas de transformação [de Queiroz and Gomes 2006]

Na aplicação da máscara o pixel a ser tratado é posicionado no centro da matriz de operação e o resultado se torna um novo pixel na mesma posição.

A par desses conceitos, o processamento se inicia transformando a imagem formada pelas bandas RGB em tons de cinza. Isso porque as variações entre as bandas serão analisadas apenas como uma variação de cinza. Segundo o site oficial do Matlab [MathWorks 2017] a transformação que a biblioteca incluída no algoritmo aplica é baseada em uma média ponderada das componentes RGB de cada pixel, dada pela fórmula a seguir:

$$F(x,y) = (0,299 * F_r(x,y)) + (0,057 * F_g(x,y)) + (0,114 * F_b(x,y)) \qquad (2)$$

Após a transformação em escalas de cinza inicia-se a aplicação dos métodos para a detecção de borda, o grupo escolheu o método criado por John Canny uma vez que os parâmetros ótimos desse processo supriam as necessidades do nosso objetivo. Esses parâmetros serão detalhados nos tópicos seguintes. A imagem que o Método resulta na saída é denominada imagem binária, isso porque os elementos variam entre 0 e 1, de forma que o 1 representa que naquele espaço há uma borda.

Figura 6. Imagem resultante do Método Canny

4.2. Método Canny para a detecção de borda

Canny apresentou a primeira ideia sobre detecção de borda em 1983 em seu artigo denominado *A variational approach to edge detection* [Canny 1983]. Nesse artigo são apresentados três parâmetros que Canny considera como sendo um filtro ideal para a detecção de borda. (1) O filtro deveria ter a menor taxa de erro possível, de forma a retornar a maior quantidade de bordas verdadeiras de uma imagem, considerando verdade uma borda proveniente da variação de cor entre um objeto e um segundo plano. Além disso, (2) a posição dos pixels marcados com variações de cores deve estar ao centro da borda. Por fim, (3) a borda deveria ter como espessura o tamanho de um pixel.

Para o retorno de uma imagem binária, representando as bordas de objetos, o método Canny realiza quatro tipos de operações sobre a fotografia original. Veja nos tópicos a seguir.

4.2.1. Suavização Gaussiana

A primeira operação aplicada sobre a imagem é a filtragem conhecida como uma Suavização Gaussiana, que terá como objetivo diminuir ruídos sobre a imagem e facilitar a detecção de bordas. Por ruídos, entende-se como sendo frequências muito altas de variações de cores. A suavização se baseia em tirar a média ponderada da intensidade de cada pixel com os seus vizinhos. Diferente da transformação em escala de cinza que tirava a média ponderada da escala RGB do *pixel*.

4.2.2. Convolução

Na próxima etapa, aplica-se a ideia de convolução sobre a imagem, de forma que uma matriz de operação 3x3 filtra *pixel* a *pixel* e encontra uma relação entre o elemento e a intensidade de variação dos seus oito vizinhos [Do Vale and DAL POZ 2002], levando em consideração pesos para cada um desses, o que equivale a primeira derivada.

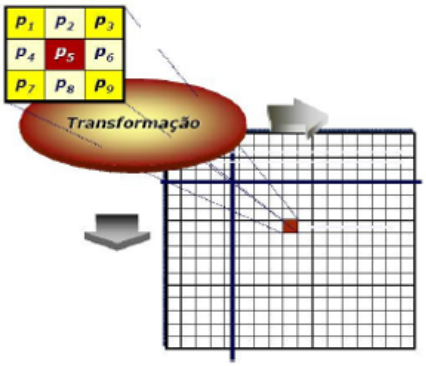

Figura 7. Exemplo da aplicação da convolução sobre uma imagem [Morgan et al. 2008]

As imagens são formadas por duas dimensões, dessa forma para melhorar o processo de encontrar a variação da intensidade de cor se aplicar a convolução nos dois eixos, x e y, simulando uma derivada parcial sobre a imagem [Biasi et al. 2002]. O representante dessas derivadas será denominado gradiente, calculado como sendo o vetor resultante das derivadas em um certo *pixel*. O gradiente pode ser dado por:

$$\bigtriangledown A(x,y) = \frac{dA}{dx}, \frac{dA}{dy} \tag{3}$$

Essa fórmula apresenta o gradiente no *pixel* A(x,y) com componente x igual à derivada do pixel em função do eixo x e y igual a derivada do mesmo pixel em função de y. O tamanho do gradiente identifica a intensidade da derivada sobre um elemento na imagem, além disso, o seu sentido e direção aponta para onde há a maior variação de cor [de Queiroz and Gomes 2006].

$$\mid G \mid = \sqrt{Gx^2 + Gy^2} \tag{4}$$

$$\theta = \arctan \frac{Gy}{Gx} \tag{5}$$

É importante ressaltar que como tratamos cada pixel da imagem como um elemento de uma matriz, o ângulo do gradiente é limitado em 0, 45, 90 e 135, considerando apenas o primeiro e segundo quadrante. Dessa forma, quando θ resulta em um ângulo diferente desses há um arredondamento, isso pode ser visto na imagem 8.

Existem diversos tipos de operadores para a detecção de gradiente, o que o grupo utiliza é o operador de Sobel que confere maior pesos aos vizinhos não nulos mais próximos, produzindo assim bordas diagonais menos intensas [de Queiroz and Gomes 2006].

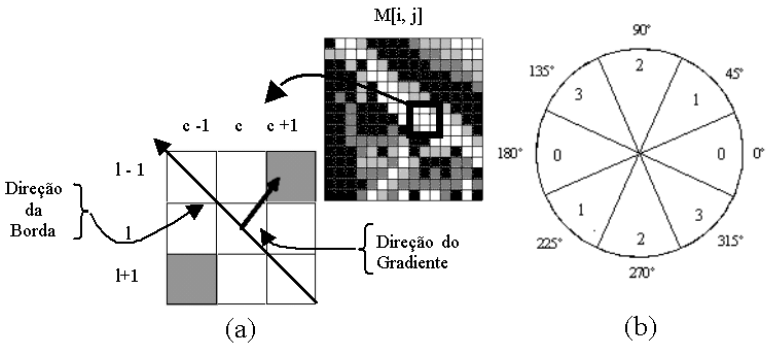

Figura 8. Direção do gradiente [Do Vale and DAL POZ 2002]

$$\Delta x = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \qquad \Delta y = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

Figura 9. Operador de convolução Sobel [de Queiroz and Gomes 2006]

4.2.3. Supressão do não máximo

Após aplicar a convolução em toda a imagem se inicia o processo de filtrar os pontos que serão bordas. Inicialmente é feito supressão do não máximo, que zera todos os valores não máximos na direção do gradiente. Relembrando que os gradientes máximos serão os pontos onde houve maior variação na intensidade da cor em uma imagem. Essa supressão possibilita que a borda seja representada apenas por um único *pixel*.

4.2.4. Limiarização

Por fim, há a limiarização, que segundo [Silva 2000] são parâmetros que delimitam a atuação de procedimentos sobre a imagem. No contexto do grupo a limiarização define um intervalo de valores que são considerados como bordas na imagem.

Entretanto, um dos principais problemas é definir os valores que serviram de parâmetros, uma vez que determinadas variáveis podem interferir na qualidade da borda, como por exemplo, iluminação, qualidade da imagem e a distância do objeto até a câmera. Isto significa que, quando um mesmo sistema é colocado em ambientes diferentes a limiarização tende a não fornecer resultados similares [Silva 2000].

Após definir o intervalo acontece o processo conhecido com binarização da imagem, definido por [Silva 2000]. Nessa etapa que há a separação do plano de fundo e do objeto, de forma que todos os valores diferentes de zero depois do processamento são considerados como borda e recebem o valor *true* (1), e o restante *false* (0).

Segundo segmentação [Morgan et al. 2008] matematicamente a detecção de borda pode ser definida como

$$G(x,y) = \begin{cases} 1 & f(x,y) > T \\ 0 & f(x,y) \leq T \end{cases} \tag{6}$$

Onde f(x,y) é o pixel na posição x e y da imagem original, T é o parâmetro e G(x,y) é o pixel da imagem binária.

4.3. Biblioteca para o processamento de imagem: OpenCV

O OpenCV (*Open Source Computer Vision Library*) pode ser utilizada em interfaces C++, Phyton e Java e é disponível para uso em Windows, Linux, Mac OS, iOS e Android. Sendo projetada para eficiência computacional, seu foco é trabalhar com aplicações em tempo real.

> A biblioteca OpenCV foi desenvolvida pela Intel e possui mais de 500 funções [4]. Foi idealizada com o objetivo de tornar a visão computacional acessível a usuários e programadores em áreas tais como a interação humano-computador em tempo real e a robótica. A biblioteca está disponível com o código fonte e os executáveis (binários) otimizados para os processadores Intel. Um programa OpenCV, ao ser executado, invoca automaticamente uma DLL (Dynamic Linked Library) que detecta o tipo de processador e carrega, por sua vez, a DLL otimizada para este. Juntamente com o pacote OpenCV é oferecida a biblioteca IPL (Image Processing Library), da qual a OpenCV depende parcialmente, além de documentação e um conjunto de códigos exemplos. [Marengoni and Stringhini 2009]

Existem cinco grupos de funções para essa biblioteca, sendo eles: Processamento de imagens; Análise estrutural; Análise de movimento e rastreamento de objetos; Reconhecimento de padrões e Calibração de câmera e reconstrução 3D [Marengoni and Stringhini 2009].

Como a OpenCV trabalha com a visão computacional, têm-se diversas maneiras de se trabalhar com imagens. Em alguns casos analisa-se um quadro fixo carregado de algum lugar. Em outros trabalha-se com *streaming* (modo ininterrupto de captar imagens de uma câmera) com dados em tempo real de algum dispositivo ou câmera. Na biblioteca mais especificamente, na parte HighGUI, é fornecida uma solução para essa situação [Bradski and Kaehler 2008].

O HighGUI pode ser dividido em três partes: hardware, sistema de arquivo e GUI. A parte do hardware está relacionada com o funcionamento das câmeras. O HighGUI permite uma maneira fácil de consultar recuperar a imagem mais recente da câmera. A parte do sistema de arquivo carrega e salva imagens. Uma característica da biblioteca é que se podem ler vídeos usando os mesmos métodos que se utiliza para ler uma câmera. Portanto se abstrai do dispositivo utilizado e continua-se escrevendo um código interessante. A parte GUI é o sistema de janelas que fornece funções que permitem abrir uma janela e abrir uma imagem dentro dessa janela. Com isso é permitido responder a comandos com o mouse e teclado [Bradski and Kaehler 2008].

5. Metodologia

Como apresentado, o objetivo geral deste trabalho é demonstrar a aplicação do cálculo de derivadas na área da Engenharia de Computação, de forma a apresentar uma aplicação prática da diferenciação. Com esse intuito, foi realizada uma pesquisa experimental para aplicar o conhecimento adquirido através da literatura baseada em processamento de imagens. Além disso, um dos objetivos específicos foi desenvolver um algoritmo capaz de encontrar as bordas de um objeto em uma imagem e com isso retornar a medição de sua altura, demonstrando de forma simplista uma aplicação dos dados extraídos de uma imagem processada.

A detecção da variação de intensidade em imagens é algo canônico na visão computacional [Canny 1986], e é baseado nos estudos do método aplicado por John Canny para detecção de bordas que foram realizadas as experimentações. Retomando o objetivo geral do trabalho, o entendimento do método Canny foi necessário, dado que este se baseia na aplicação da primeira derivada da função Gaussiana.

Inicialmente os primeiros algoritmos foram desenvolvidos no software Scilab, pois além de gratuito, possui uma maneira simples para demonstrações matriciais e gráficas. Este software integra cálculo com matrizes, construção de gráficos e análises numéricas em um ambiente de fácil manipulação. A sua utilização foi essencial para o desenvolvimento do trabalho, pois simplificou a visualização da maneira que uma imagem é interpretada numericamente por um computador.

A primeira etapa foi dedicada para implementação de um algoritmo que através da manipulação da matriz de uma imagem, retornasse uma matriz resultante apenas com elementos binários. Utilizando uma biblioteca própria para processamento de imagens no software Scilab, foi desenvolvido um algoritmo que recebe uma imagem e a salva na forma matricial, onde cada *pixel* representa um elemento da matriz. O número atribuído a esse pixel, ou seja, ao elemento dessa matriz, é um valor do sistema RGB. Na sequência, utiliza-se um comando para transformar o valor RGB em um representante na escala de cinza, através da média ponderada entre as intensidades de vermelho, verde e azul. Essa modificação é necessária para diminuir as variações entre as cores dos *pixels*. Por fim, é aplicado o método Canny, que é capaz de identificar mudanças na intensidade dos *pixels* da imagem para detectar a presença de bordas.

Após a demonstração das formas matriciais que uma imagem assume em um computador, seja em cores ou na escala de cinza, espera-se que os alunos sejam capazes de perceber uma das diversas aplicações possíveis do cálculo de derivada para encontrar as bordas presentes na imagem.

Ficou decidido durante a realização do projeto a substituição do software Scilab por um software capaz de copilar a linguagem de programação C++. Essa decisão foi tomada pelo fato de essa linguagem trabalhar com a biblioteca OpenCV. Como explicado, essa biblioteca possui eficiência para aplicações em tempo real e, além disso, a linguagem C++ demanda menos processamento e é orientada à objeto o que amplia as possibilidades a serem trabalhadas.

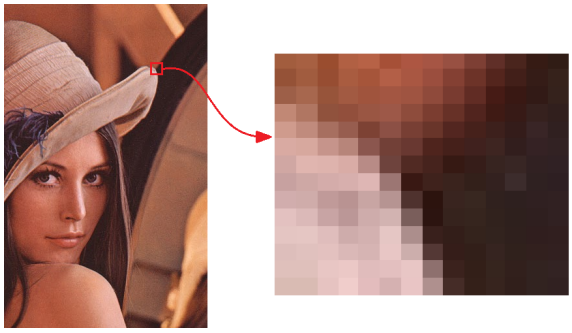

Figura 10. Detalhe de uma imagem para a visualização dos pixels

162	167	154	177	182	166	173	164	152	123	96	74	54	48
149	161	151	169	166	179	169	155	132	109	84	66	53	54
157	167	151	162	160	163	159	143	122	97	70	57	50	53
193	176	144	146	154	157	142	123	103	81	63	55	54	49
207	192	167	151	125	140	120	105	83	70	61	52	57	48
217	215	198	183	139	114	105	79	69	64	62	50	53	50
208	224	222	217	184	110	81	70	54	54	55	56	49	48
199	206	212	215	222	156	57	63	58	51	51	60	47	46
210	207	192	185	200	182	79	44	57	56	48	46	48	49
227	219	200	187	206	210	122	44	54	54	49	46	47	49
228	229	223	206	211	229	172	57	51	52	50	47	48	49
222	224	227	218	214	232	204	84	53	50	54	48	50	50
223	223	232	235	224	227	232	142	59	51	54	50	50	51
217	217	230	240	227	213	233	175	69	52	55	49	50	49

Matriz RGB

77	97	98	82	90	89	84	67	50	39	28	28
77	92	86	97	90	82	68	54	40	33	29	34
84	91	86	89	87	77	64	48	30	26	28	34
83	81	88	91	79	64	52	39	28	28	34	32
113	95	69	84	66	56	41	35	34	30	40	34
151	135	90	67	59	39	34	36	39	32	37	36
182	175	143	70	44	37	27	31	37	39	35	36
176	179	186	121	26	36	36	32	34	46	35	34
158	151	168	151	52	20	37	39	34	33	34	35
168	155	176	182	96	22	35	37	34	33	33	35
193	176	181	201	145	34	31	35	35	33	32	33
196	188	184	202	177	58	31	31	37	33	33	34
201	203	192	195	202	115	35	31	37	33	36	37
198	208	193	181	201	147	45	32	36	32	36	35

Matriz em escala de cinzas

Figura 11. Matriz representante do recorte da figura 10

0	0	0	0	0	0	0	0	0	**1**	0	0	0	0
0	0	0	0	0	0	0	0	0	**1**	0	0	0	0
0	0	0	0	0	0	0	0	**1**	0	0	0	0	0
1	**1**	0	0	0	0	0	**1**	**1**	0	0	0	0	0
0	**1**	**1**	0	0	0	**1**	**1**	0	0	0	0	0	0
0	0	**1**	**1**	**1**	0	**1**	0	0	0	0	0	0	0
0	0	0	0	**1**	**1**	0	0	0	0	0	0	0	0
0	0	0	0	0	**1**	0	0	0	0	0	0	0	0
0	0	0	0	0	0	**1**	0	0	0	0	0	0	0
0	0	0	0	0	0	**1**	0	0	0	0	0	0	0
0	0	0	0	0	0	**1**	0	0	0	0	0	0	0
0	0	0	0	0	0	0	**1**	0	0	0	0	0	0
0	0	0	0	0	0	0	**1**	0	0	0	0	0	0
0	0	0	0	0	0	0	**1**	0	0	0	0	0	0

Matriz binária

Figura 12. Matriz binaria do recorte da figura 10

6. Resultados

Retomando um dos objetivos específicos deste trabalho, utilizamos a matriz binária gerada para realizar a medição de um objeto contido na imagem. A matriz resultante da aplicação do método Canny é uma matriz contendo apenas zeros e uns, onde zero repre-

senta a ausência de borda e um a presença. Sabendo disso, foi criada uma função que retornava a primeira e a última linha da matriz que contém o elemento um, ou seja, o elemento que indica a primeira e última borda do objeto. Com essas informações foi possível, através de subtração, encontrar a altura do objeto dada em *pixels*.

Foram realizadas diversas experimentações para constatar a correlação entre a altura em *pixels*, e a altura real do objeto em milímetros. Para esse processo, foi necessário o controle de algumas variáveis, como: a distância do objeto à câmera; a rotação do objeto; e o processo de limiarização do método Canny.

Objeto	Tamanho (mm)	Medida em pixels	Relação Pixel/mm
A4 horizontal	210	280	0,75
A4 vertical	294	394	0,75
Celular	140	185	0,76
Cartão vertical	94	124	0,76
Cartão horizontal	66	88	0,75

Média:	0,75
Desv. Padrão:	0,005

Tabela 1. Resultado das medições realizadas à 500 milímetros de distância.

Objeto	Tamanho (mm)	Medida em pixels	Relação Pixel/mm
A4 horizontal	210	191	1,10
A4 vertical	294	269	1,09
Celular	140	126	1,11
Cartão vertical	94	86	1,09
Cartão horizontal	66	61	1,08

Média:	1,10
Desv. Padrão:	0,011

Tabela 2. Resultado das medições realizadas à 700 milímetros de distância.

Objeto	Tamanho (mm)	Medida em pixels	Relação Pixel/mm
A4 horizontal	210	142	1,48
A4 vertical	294	188	1,56
Celular	140	91	1,54
Cartão vertical	94	64	1,47
Cartão horizontal	66	45	1,47

Média:	1,49
Desv. Padrão:	0,045

Tabela 3. Resultado das medições realizadas à 1000 milímetros de distância.

Como pode ser observado na Tabela 1, o desvio padrão da relação pixel/milímetros é bem inferior àquela constatada na Tabela 3. Isso indica que quanto mais próximo a

câmera estiver do objeto medido, maior será a repetitividade do processo, assim como melhor será a precisão. Isso está relacionado com a perda de detalhes do objeto ao afastá-lo da câmera. Em posse dessas informações, foi decidido realizar a medição de objetos em tempo real, e não mais utilizar imagens estáticas. Para esse processo, foi necessário criar uma etapa para estabelecer uma relação inicial entre *pixels* e centímetros para ser mantida durante a medição de novos objetos. Este processo consiste na utilização de um objeto com dimensões conhecidas, e a partir da medição deste objeto dada em *pixels* computar a relação entre pixel e centímetro a ser mantida.

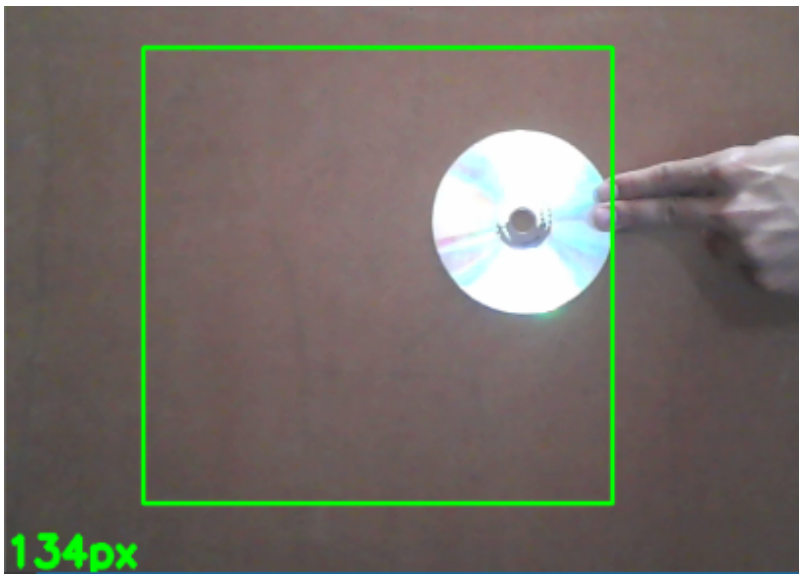

Figura 13. Computando altura em pixels para ser utilizado como referência

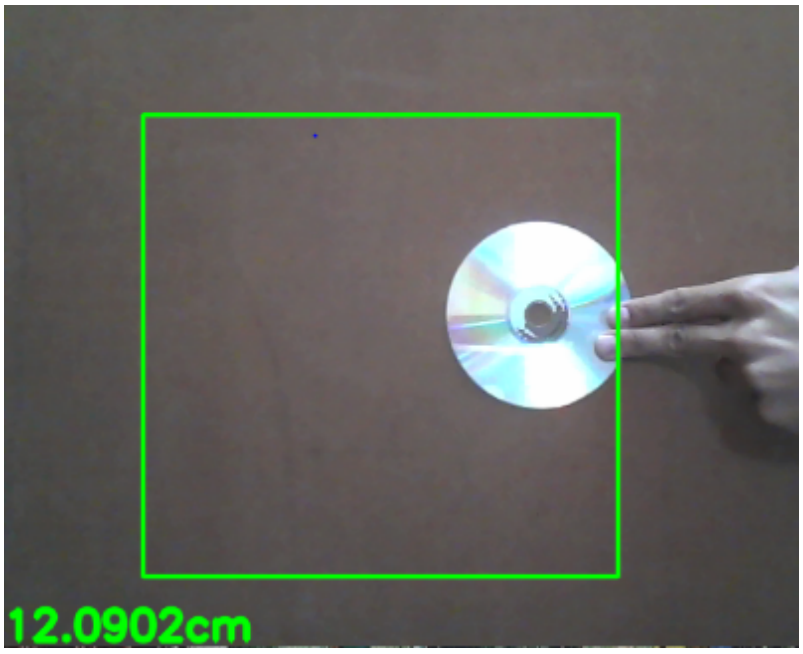

Figura 14. Criando relação cm/px

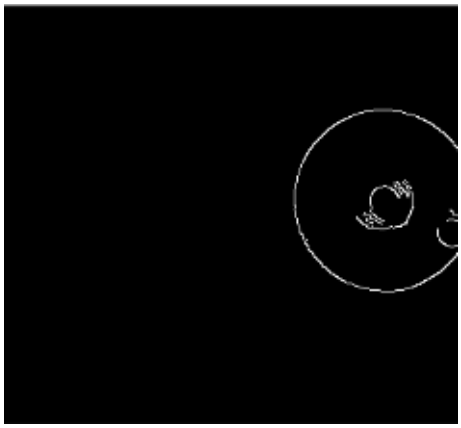

Figura 15. Bordas extraídas de um objeto

Na figura 13, o diâmetro do CD foi equivalente à 134 pixels. Como um CD tem diâmetro de 12,0 centímetros, o software computou a relação pixel/centímetro equivalente à aproximadamente 0,089 cm/px. Isto significa que cada pixel equivale a 0,089 centímetro. Esta referenciação foi realizada com a câmera a uma distância de 750 milímetros do objeto, e mantida para as demais medições.

Como as bordas que não representam o objeto, e as bordas do objeto são armazenadas da mesma maneira, para realizar as medições era necessário criar um ambiente em que as únicas bordas presentes fossem às do objeto desejado. Essa situação era devido ao fato de o algoritmo procurar o primeiro e o último elemento da matriz que representasse borda, e quando encontrado falsas bordas, ou bordas que não representam o objeto, falhas são geradas. Percebido isso, foi desenvolvido uma nova função no programa que é responsável por delimitar a área de interesse. Esta função tem como simples objetivo reduzir a área de busca por bordas, transformando a matriz da imagem em uma matriz menor em que objeto se encontra inserido.

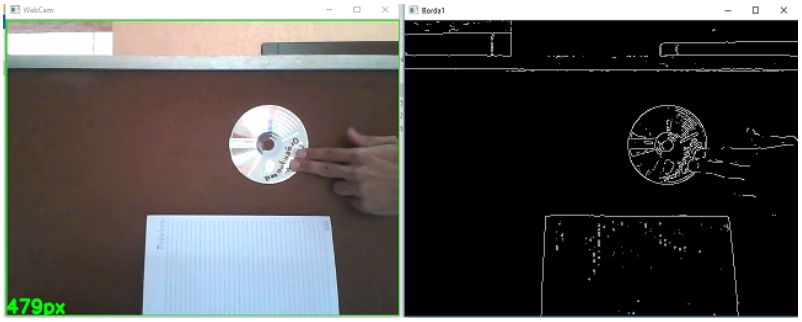

Figura 16. Detecção de bordas sem delimitação de área

Como pode ser observado na figura 16, onde não há delimitação da área de busca, o algoritmo retorna uma altura em pixels equivalente à 479px. Já na figura 17, a região de busca por bordas corresponde à área interna do retângulo de contorno verde, isso significa que somente o que estiver compreendido na figura 18 será analisado.

Além do problema com a área de interesse, houve também a necessidade de permitir que o usuário pudesse parametrizar, quando necessário, a limiarização desejada para

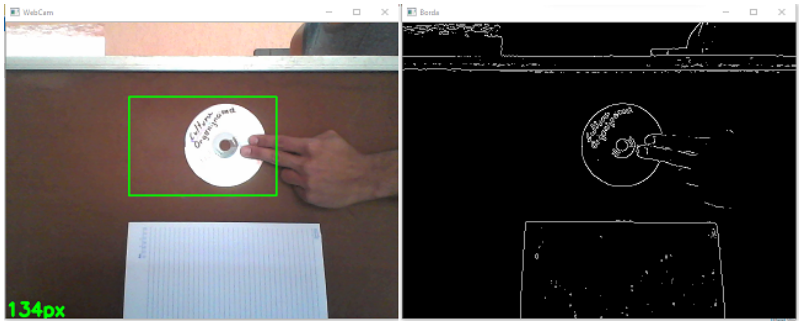

Figura 17. Delimitação da área de busca por bordas do objeto

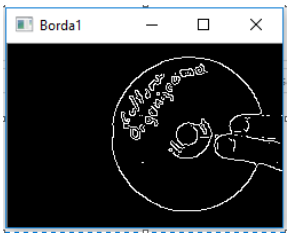

Figura 18. Imagem representante da área delimitada na figura 17

o processo. Pois, a aplicação da limiarização é difícil e envolve experimentação. “A permanência de falsas bordas, após a limiarização, pode ter como motivo a escolha de um limiar baixo, ou alto demais” [Do Vale and DAL POZ 2002, p. 9]. Na figura 19 é possível perceber o impacto de uma limiarização mal aplicada. Na imagem à esquerda as bordas ficam restritas ao objeto, já na imagem à direita, ocorre a presença de falsas bordas, o que interfere na medição.

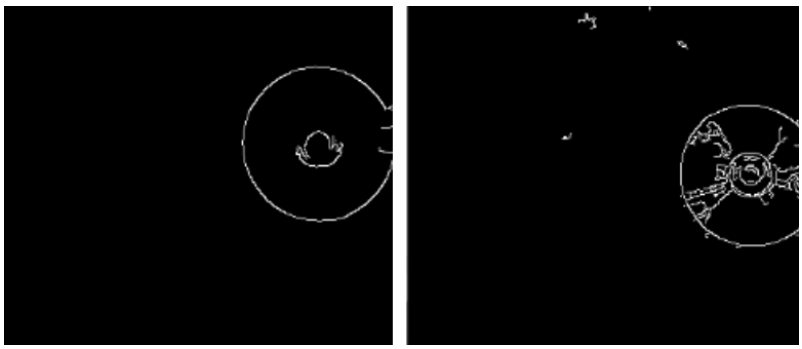

Figura 19. Diferenças de limiarização e presença de falsas bordas

Por fim, reduzindo a área de busca, e ajustando a limiarização durante o processo de medição, o algoritmo foi capaz de retornar medições dos objetos com uma precisão de 0,0975 centímetro, ou seja, inferior à 1 milímetro (vide Tabela 4), o que consiste em um resultado aceitável, visto que era buscado a demonstração de um processo possível. Porém, o fato de trabalharmos com uma câmera 2D, implica na perda de percepção de profundidade. Logo, quando há rotação do objeto a medição passa a se invalidar. Aprofundar os cálculos para encontrar medições do objeto mesmo com rotações, é uma das possíveis vertentes que este trabalho poderá seguir, portanto não será abordado essa variável de processo.

Objeto	Esperado (cm)	Medido (cm)	Erro (cm)
CD Padrão	12,00	12,09	0,09
Mini CD	8,00	8,03	0,03
CC Horizontal	5,40	5,50	0,10
CC Vertical	8,56	8,66	0,10
A4 Horizontal	21,00	21,11	0,11
A4 Vertical	29,70	29,77	0,07
Cartão de visita Horizontal	4,80	4,96	0,16
Cartão de visita Vertical	8,90	9,02	0,12
		Média	0,0975

Tabela 4. Resultados das medições realizadas

7. Considerações Finais

As imagens transmitem diversas informações sobre o mundo, como fatores históricos e culturais de uma determinada sociedade, noções biológicas de um ser vivo utilizando o raio – x, ou ainda informações da estrutura física de determinados objetos. O intuito desse trabalho era entender como a derivada poderia auxiliar no estudo dessa última informação, esclarecendo para os alunos do primeiro ano a aplicabilidade do Cálculo na Engenharia de Computação.

A área de foco foi o processamento de imagem, mas especificamente a detecção de bordas utilizando o método de Canny. Essa área possuía vantagens que possibilitava visualizar a derivada e exigia pouco conhecimento para entender, de forma básica, como se produzia uma imagem de bordas como saída do processamento. Além disso, outra razão pela escolha dessa área, foi a possibilidade de visualizar o processamento de imagens capturadas pela Webcam, o que possibilitariam aos alunos avaliar, por meio de algoritmos, o ambiente ao seu redor em tempo real. A análise proposta por esse trabalho foi utilizar o método Canny para medir objetos.

A principal dificuldade encontrada foi conseguir filtrar as informações necessárias para se ter uma base do processamento sem que fosse exigido muito conhecimento. Além disso, o grupo entrou em contato com dificuldades existentes na área de computação visual, como: ruídos na imagem, bordas não desejadas, relação entre o tamanho do objeto e sua distância da câmera e etc.

Portanto, por meio desse trabalho foi possível constatar que o processamento de imagem é uma área promissora que integra os conceitos do Cálculo da derivada e a área da computação. Como estudos futuros, o grupo pretende aperfeiçoar as técnicas de processamento utilizadas nesse trabalho, embarcando conceitos de outros períodos, além do primeiro e segundo. As melhorias que o grupo almeja são (1) conseguir fazer medições podendo varia a rotação do corpo e a distância desse até a câmera sem a necessidade de configurar uma nova relação; (2) automatizar a limiarização de bordas para cada ambiente; por fim (3) otimizar as estruturas do algoritmo.

Referências

[Biasi et al. 2002] Biasi, H. H. D. et al. (2002). Desenvolvimento de uma metodologia de visão computacional para o auxílio no planejamento cirúrgico no implante de próteses endoluminais.

[Bradski and Kaehler 2008] Bradski, G. and Kaehler, A. (2008). *Learning OpenCV: Computer vision with the OpenCV library*. "O'Reilly Media, Inc.".

[Canny 1986] Canny, J. (1986). A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence*, (6):679–698.

[Canny 1983] Canny, J. F. (1983). A variational approach to edge detection. In *AAAI*, volume 1983, pages 54–58.

[de Queiroz and Gomes 2006] de Queiroz, J. E. R. and Gomes, H. M. (2006). Introdução ao processamento digital de imagens. *RITA*, 13(2):11–42.

[Dias 2016] Dias, G. A. (2016). Cálculo diferencial e integral e suas aplicações.

[Do Vale and DAL POZ 2002] Do Vale, G. M. and DAL POZ, A. P. (2002). Processo de detecção de bordas de canny. *Boletim de Ciências Geodésicas*, 8(2).

[Marengoni and Stringhini 2009] Marengoni, M. and Stringhini, S. (2009). Tutorial: Introdução à visão computacional usando opencv. *Revista de Informática Teórica e Aplicada*, 16(1):125–160.

[MathWorks 2017] MathWorks (2017). Documentation — rgb2gray. Disponível em: `https://www.mathworks.com/help/matlab/ref/rgb2gray.html`.

[Morgan et al. 2008] Morgan, J. et al. (2008). Técnicas de segmentação de imagens na geração de programas para máquinas de comando numérico.

[Silva 2000] Silva, L. (2000). *Segmentação de Imagens de Profundidade por Detecção de Bordas*. PhD thesis, Dissertação (Mestrado), UFPR, Curitiba, 2000. 113p.